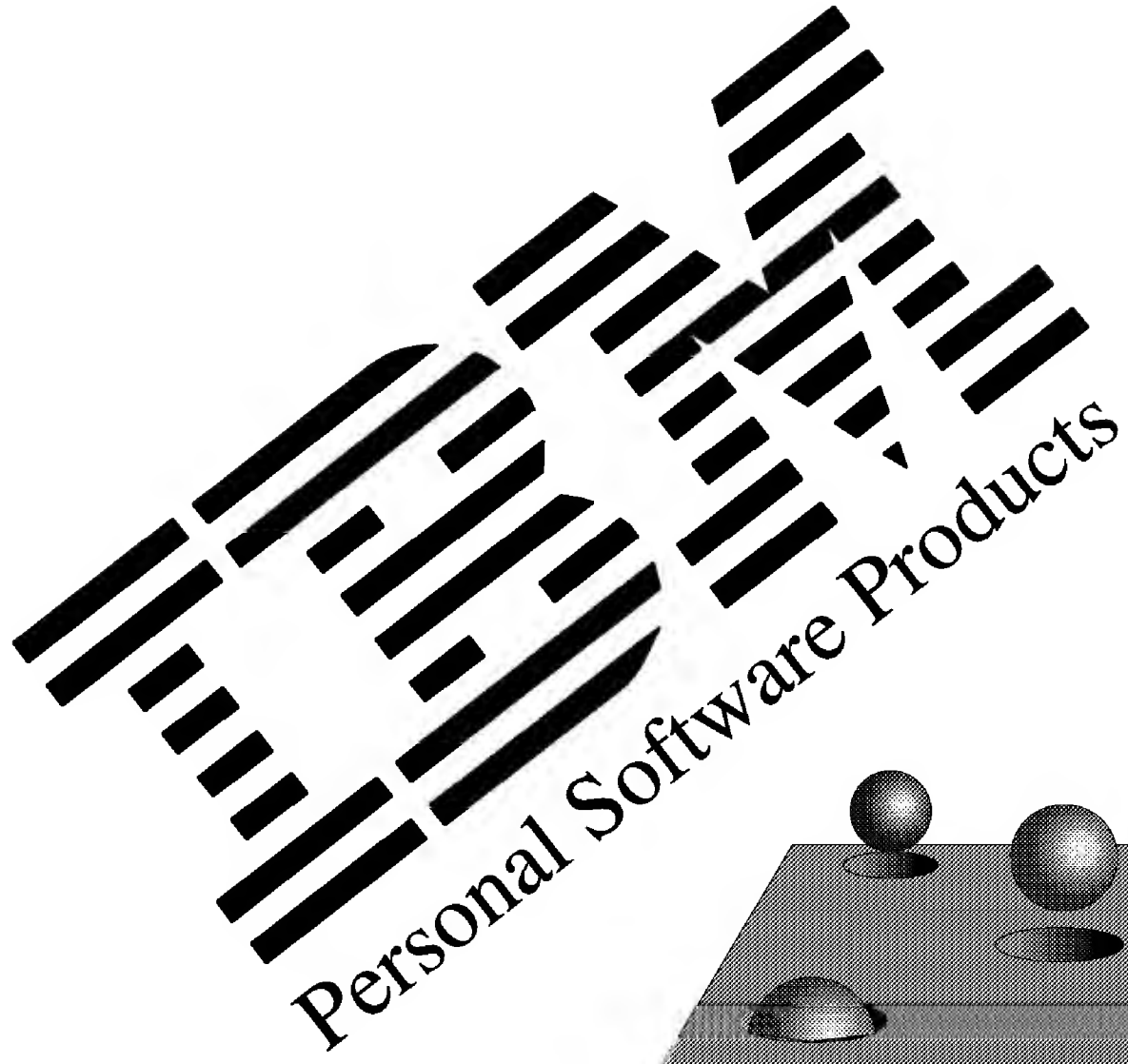
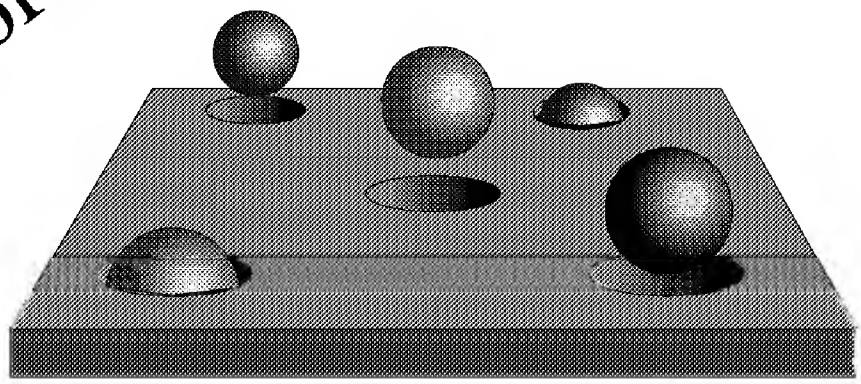




Transforming Your Business With Object Technology

Strategy for Object Technology

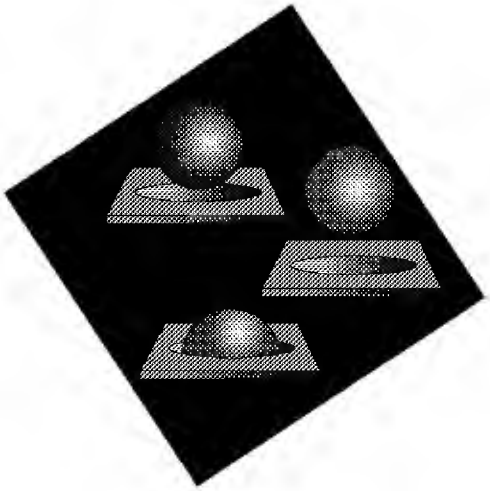
IBM's Personal Software Products' strategy for object technology is straightforward: to play a leadership role in the development of an object technology environment that will change the economics and to simplify the process of software development for our customers.

Our efforts are guided by four underlying principles:

- We will take a leadership role in research and development.
- We will participate actively in the development and adoption of industry standards.
- We will complement the work of other leading object technology companies through key alliances and partnerships.
- We will simplify the integration of object technology into our customers' operations.

November 1993

Table of Contents



1: Changes in Business Require Changes in Technology

1: Summary

- Today's dynamic business environment is propelling object technology (OT) into the computing mainstream.
- Corporations are moving to flatter, horizontal organizations that are process and work-flow driven.
- Companies will emphasize core competencies as the *virtual corporation* takes hold.
- A new information technology (IT) infrastructure will be required to deliver the distributed information environment necessary to meet business needs.
- IT will be facing growing pressure to deliver powerful, distributed applications quickly.
- OT will play a major role in facilitating a new breed of applications that make businesses more competitive.

Today's dynamic business environment and key technological innovations are combining to propel object technology (OT) into the mainstream of computing. This convergence comes at a critical time – when corporations are in the midst of a significant global shift in the composition, organization and operation of their business.

Companies are re-engineering, restructuring and resizing at an unprecedented rate to become more competitive or, in some cases, to survive. Businesses are transforming themselves to become flat, networked organizations that are process and work-flow driven.

To deliver viable products and services faster, businesses are exploring new relationships – looking beyond their organization to embrace customers, suppliers and even competitors as a valuable part of the delivery mechanism. The concept can be viewed as a *virtual corporation* – a highly focused organization that emphasizes its core competencies and relies on long-term alliances and a contracted, dynamic work force tailored to fill specific needs and skills. ***(See Figure 1)***

In addition, organizational dynamics and technological innovation are speeding the transition. Computing power that once occupied entire rooms is now on the desktop, in a briefcase or on the kitchen table at home. This power and availability coupled with the information needs of a virtual corporation are forcing a major transition in the information technology (IT) infrastructure.

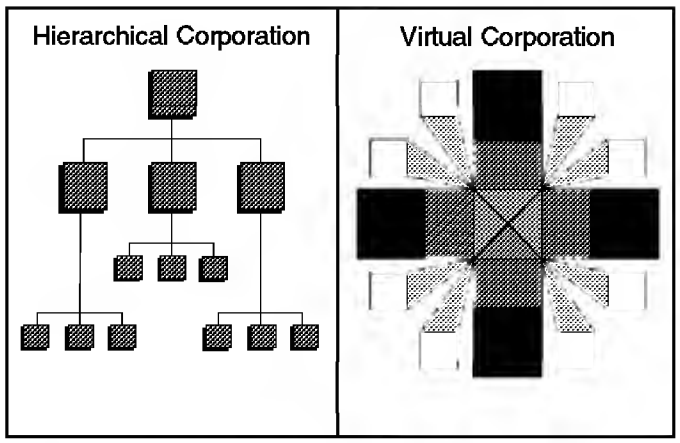


Figure 1 – *Organizations are migrating to flat, networked and work-flow driven.*

For most organizations, the days of clerical-based IT computing – simple, static, centralized information activities – passed long ago. Today, most IT organizations have reached beyond the walls of their enterprise to connect suppliers, as well as their fulfillment channels, in order to smooth product and/or service delivery and compress cycle times. While these systems will continue to play an important role, the IT infrastructure must also:

- provide real-time access to information that reinforces each user's expertise and
- adapt easily to the changing business environment – regardless of the people, locations, organizational issues and processes. ***(See Figure 2)***

Delivering this infrastructure is an enormous undertaking. To begin with, there's the fundamental complexity of the environment itself and the incompatibility of the disparate hardware and software involved. As more and more IT decisions are driven by impatient profit-motivated end-user departments, information systems (IS) organizations will face growing pressure to deliver powerful, distributed applications quickly. The ability to involve the people who understand the business problem is critical.

Object technology radically alters the way software systems are developed as well as the ways businesses use these systems to achieve competitive advantage. OT changes the economics of application development – and the possibilities.

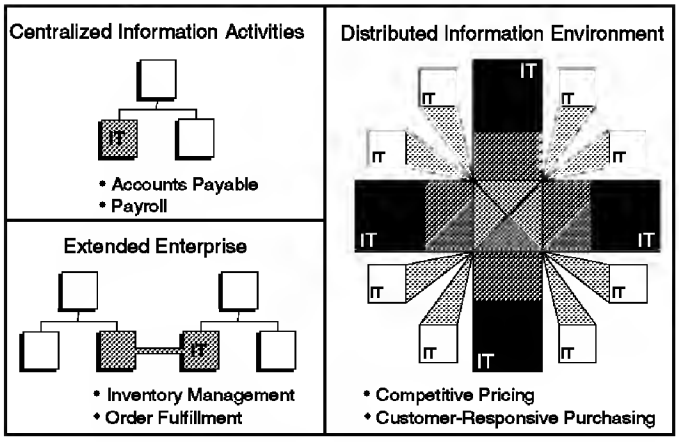


Figure 2 – *These components illustrate the distributed information environment.*

The promise of OT is compelling: to take a complex business problem and solve it by assembling and/or extending reusable software components. For example, if a bank wants to build an application that makes personal bankers more productive, a developer might start with a loan framework that models the banking business. It might contain a customer information object, a portfolio object, a credit check object and a risk analysis object. Any of these objects could be modified or changed without affecting the rest of the objects in the framework, or the ways in which they interact. So by altering only the risk assessment object, a developer could create a specialized loan application for valued customers – with an order of magnitude less effort than if he or she used a current procedural-based approach.

The benefits are evident. Since OT applications more closely model the business problem they're designed to solve, they're easier to develop and to maintain. Conversations center around loan approval or portfolio, rather than complex, procedural-based syntax. Once written, object programs are reusable and easily extended. Initial programmer effort is used over and over again. Creating a student loan offering, for example, becomes simply a matter of extending the standard loan framework, to adjust the risk analysis object and to change the amortization period and term of the loan.

However, objects themselves are not enough. They must reside in an open, distributed environment that will allow the use of components in a "plug-and-play" fashion. This enables the vision of dynamic re-engineering to become a reality. With that vision rapidly approaching, IBM Personal Software Products (PSP) is taking the lead in providing OT-based solutions, frameworks and services.

The promise of OT is compelling: to take a complex business problem and solve it by assembling and/or extending reusable software components.

The fact is that OT provides the foundation for transforming businesses as well as the catalyst for reshaping the information-processing industry itself. A growing understanding of this potential is leading to fundamental changes in all aspects of the IT industry: services, hardware, software, distribution, even terminology.

Terms and Definitions

New terms are entering the mainstream information technology vocabulary. A shared definition of the most common object-oriented (OO) terms will simplify this discussion.

A framework is a set of objects that work together to perform a specific task. Graphical user interface (GUI) frameworks, for example, let objects like windows, menus and dialogs work together to provide an efficient way to build consistent graphical applications. There are frameworks for business problems, too, such as hospital management or inventory control. Well-designed frameworks provide the general design and implementation for a specific problem and allow a developer to customize it to a particular situation.

An object (also referred to as a component) is the data and logic that represents a useful element in an application. For example, in a financial application, there may be objects that represent account, branch and customer. The customer object can be defined to perform certain functions, but the details of how it does them are hidden from all but the designer of the customer object itself. Objects are valuable in designing and implementing software because they hide complexity.

- ***A message*** is how one object requests services of another. A customer object may send the *print statement* message to a specific checking account object. The message identifies the method that the object will use to perform the request.
- ***A method*** is an operation that an object can perform. For example, an account object may have separate methods for performing a debit, a credit or printing a statement.
- ***A module*** is a loosely but widely used term that describes a piece of code that pertorms some function. Objects are sometimes referred to as self-contained modules to emphasize the notion of encapsulation. Usually an object's methods are implemented as code modules.

An Object Request Broker (ORB) is the mechanism that enables objects to communicate with each other across a network. It provides services for security, registration and object management. The Object Management Group (OMG), an industry consortium, has defined the Common Object Request Broker Architecture (CORBA) to specify standards so different ORB implementations can work together.

Encapsulation is the practice of making an object self-contained and hiding its internal structure. Thus an object maintains its own data and the logic to perform operations while exposing to other objects only what is necessary to request actions. Encapsulation leads to flexible designs, as internal structures – or even whole objects – can be modified without affecting the rest of the system.

Polymorphism (literally, many shapes) is the property that allows different objects to respond to the same request. For example, savings account, checking account and loan account objects may each handle a request to *print statement*, even though each statement may contain different information. Polymorphism reinforces the value of encapsulation in an application by reducing the amount of information we need to know about an object. This allows objects to be modified or replaced without affecting other areas of the application.

Inheritance is the mechanism that allows a developer to create a customized object based on the implementation of a general one. For example, an account object may handle debits and credits and produce statements. A savings account object may inherit all ot those features and accrue interest on the outstanding balance. Inheritance increases productivity by providing reusable software without sacrificing flexibility.

2: Objects: More than a Technology – A Solution to Critical Business Problems

2: Summary

- Businesses need flexible, adaptable information systems.
- Complexity and accelerating change are driving the industry to seek alternatives to expensive, rigid legacy systems.
- Business-driven applications can be developed faster using objects as an enabling technology.
- OT-based application development will integrate three basic steps.
- New object solutions will be hybrids of vertical and horizontal applications based on process and modeling.
- These new diagonal applications will be mission-critical yet flexible, easy to use and end-user controlled.

In today's competitive environment, businesses of all sizes are recognizing that inflexible and unresponsive systems can no longer sustain their organizations. Although the distributed client/server model has been championed as the way to address the flatter organizational structure, the complexity involved in implementing these solutions has slowed the progress. OT will help accelerate the move to this powerful, multi-platform computing environment by dramatically simplifying the development of distributed applications.

Object technology, by definition, is well-suited for distributed systems because both the business logic and the data are contained within objects, allowing them to be located anywhere within a distributed network. OT can mask or remove the majority of specific platform characteristics. The application development team can then place more emphasis on understanding the marketplace, customer interactions and business processes. OT also will enable increased user involvement for designing and enhancing applications – and perhaps even for modifying them independently.

With the advent of open, distributed objects, the end user and IT consultant will participate jointly in designing a solution for a set of business problems. First, the business process description becomes the foundation for the solutions. Then building on an industry framework the development process can be iterative, resulting in an operational prototype more quickly. Since the solution is expressed in process/simulation terms, the end user can be directly involved in enhancing the solution. The second step begins an iterative process of coding, testing and redesigning using additional class libraries of business objects. The third step is maintenance by the end user through enhancement or redesign of the solution on a dynamic basis to meet changing business needs.

As object-oriented (OO) solutions evolve, they will look less like traditional applications and more like intelligent, interactive business processes. Today, end users experience the procedural world of information processing applications on two basic axes: vertical mission-critical/industry-specific applications and horizontal personal-productivity applications. The user has limited involvement and control of the flexibility and usability of vertical applications, while exercising extensive control over the personal productivity applications he or she chooses to use. OT will enable vertical and horizontal applications to evolve to a new generation of diagonal applications – mission critical yet heavily influenced and directed by the user's need for flexibility and ease of use.

3: Market Expectations for Object Adoption

3: Summary

- Object technology adoption is inevitable.
- The real question isn't "if" – but rather "when."

Studies have shown that market and technology forces make the adoption of OT inevitable. ***(See Figure 3)***

Markets are moving toward shorter product life cycles and placing recognized value closer to the customer. Though software has made fundamental improvements in function, ease of use and quality, it has not kept pace with hardware's move toward smaller, cheaper and faster units. Objects are the software equivalent of the microprocessors that triggered a radical change in the computer hardware industry business model.

Objects are the software equivalent of the microprocessors that triggered a radical change in the computer hardware industry business model.

Companies are moving toward OT for two reasons: first, to reduce the time required to develop applications; and second, to design applications that create competitive advantages by improving responsiveness to shifts within their industry. There is growing evidence of a strong market pull for OT from corporations developing complex mission-critical applications. Existing procedural-based technology has reached its limits for many of these firms. In addition, many companies that traditionally have been conservative with their adoption of technology are realizing they, too, have reached the limits of traditional computing approaches. They are now accepting OT as the way to rebuild their software systems.

A recent International Data Corporation (IDC) study of 800 representative customers reinforced this trend. The study indicates that 12% of surveyed customers already are using OT extensively, 44% are exploring OT benefits, 73% are moving to OT from an existing client/server environment and 68% are moving to a distributed systems environment while using and/or exploring OT. All of these customers recognize the potential OT offers.

A research study by Workgroup Technologies, Inc. shows that a variety of organizations, lines of business within large corporations, companies and educational institutions have already begun deploying OT applications.

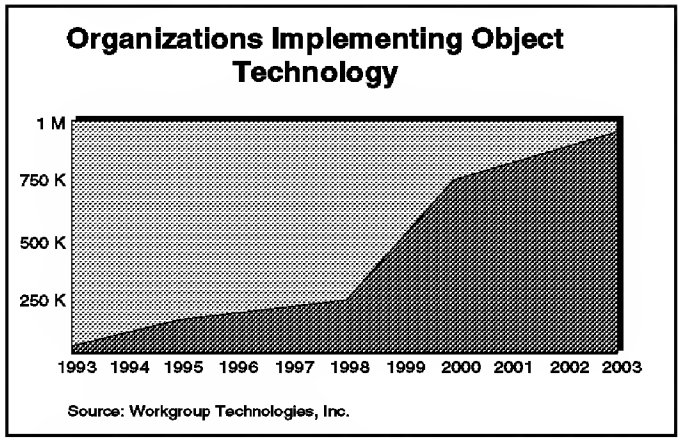


Figure 3 – For many companies, the inevitable move to OT already has begun.

And the number of advanced tools becoming available over the next five years will fuel the acceptance of OT.

OT offers the basis for reusable, "plug-and-play" software that will complement the new infrastructure required to optimize business processes and applications on a dynamic basis. Its principles allow generalized components to be used as is, or more important, to be specialized for a particular implementation. OT enables components to be connected as application and system frameworks to solve complex, mission-critical problems. Despite the potential of OT, a number of critical inhibitors have preempted its widespread deployment.

4: Objects – Why Now?

4: Summary

- Object technology has overcome 25 years of obstacles to acceptance.
- The formation of the Object Management Group has generated standards for OT components.
- Development of the Object Management Architecture has provided definitions for the primary segments of any OO environment.
- Movement of OT into the mainstream has persuaded cautious companies to invest in this technology.

Given the obvious potential of OT, it's important to understand the problems the industry has had to overcome, as well as the changes that have taken place, in order to finally make OT a viable option today.

Problem: *Component incompatibility*

- Despite today's choices of OT languages, components generated by different tools do not work together. Not only is it difficult to use the components outside of a particular domain, it is impossible to customize them without having the source code or operating within the system that created them. For example, if a generalized component is created in C++, it is difficult to use it and impossible to refine it from Smalltalk or COBOL.

Solutions: *Emergence of industry standards*

- To address the issues of compatibility and object standardization, the Object Management Group (OMG) was formed in April 1989. This non-profit, international corporation is dedicated to establishing industry guidelines and object-management specifications that will provide a common framework for distributed application development.

  Standardization will allow components to be packaged in binary form which, in turn, will allow them to be invoked and refined from any supported language, e.g., C++, Smalltalk or Object COBOL. In addition, a logical extension of this cross-language standard is a set of services that will allow components to be located and managed across a network. The ultimate goal is to automate these networks and other functions so the developer need only concentrate on application functionality.

  The OMG has developed the Object Management Architecture (OMA) as the basis for defining the primary pieces of any object-oriented computing environment. The OMA defines the Object Request Broker (ORB) as the mechanism that sends and receives messages between objects. The ORB is central to distributed OT because it provides an environment where different applications running on different computers in different locations and environments can interoperate seamlessly.

  The leading standard defined by OMG is the Common Object Request Broker Architecture (CORBA). CORBA is a specification that defines ORB implementations, services and interfaces. While CORBA is neither perfect nor complete, the OMG now has more than 300 members, and OT vendors are working to be labeled CORBA-compliant. For customers, CORBA compliance provides assurance that the language, tool or implementation selected provides the minimum amount of acceptable standardization necessary to bridge incompatible technologies.

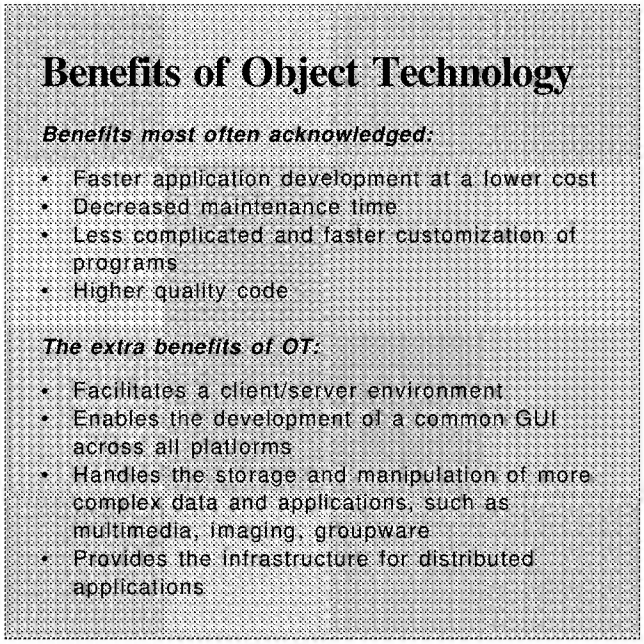
Problem: *Companies have been reluctant to build bet-the-business applications based on niche technology*

- Although the promise has been appealing, the fact is that corporate IS organizations are, by necessity, very conservative. Thus, the majority of OT choices have come from small, relatively young organizations with highly specialized solutions that address highly specialized issues. So often, when the Advanced But Very Small OT Company visits the Very Large and Conservative IS Shop, it encounters a very conscientious group of developers who would rather resort to proven approaches than risk their careers on an unknown company – regardless of the technology's potential.

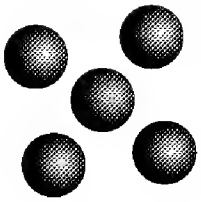
Solution: *Object technology is entering the mainstream*

- Standards bodies, consortia and large systems vendors are entering the market – all signs that OT is here to stay. Over time, this will help address other issues such as the small pool of qualified and skilled designers, architects, programmers and assemblers; the general scarcity of standardized developer toolkits, frameworks and class libraries; and the need for education sources, consulting and contract engineering services.

IBM PSP recognized that OT is strategic to the future of the computing industry, thus we are playing a leading role in its advancement – not only through development of our own technology, but through involvement with consortia and other key industry vendors.



A look into the future



The year is 1998 and the sales manager for the western region of a large grocery store chain is opening a new store in a very competitive market. Although his new location is convenient, the manager knows his customers are very price conscious and, to make a reasonable return, he must control inventory closely. His initial plan is to exploit the infrastructure already in place:

- *point-of-sale registers that read bar codes, look up prices and adjust store inventory records, and*
- *warehouse systems that track orders and deliveries to and from major suppliers nationally as well as locally.*

Although these tools provided a competitive advantage when they were installed originally, the sales manager knows that today, even the family-owned grocery store down the street is using the same technology. So he's preparing to turn up the heat by:

- *adding "price scouts" who use mobile devices to log competitive prices and radio them into a central system, and*
- *developing a customer information system that will track customer preferences to help guide future purchasing of higher-margin items such as pre-prepared delicatessen foods.*

It took years to design, build, test, debug and then roll out the first retail store systems – especially those designed for just-in-time delivery. Issues such as compatibility with existing systems hardware, systems software and networks had to be negotiated by a systems integrator. Then custom applications had to be developed and tested. Once the system was designed, adding a new store still meant extending the network, updating tables and adding code to recognize the new store in countless applications.

With the full deployment of OT and an industry full of compatible, standards-based objects and frameworks, designing this new system won't be nearly as difficult. The radio-based personal communicator will come with software that registers the device according to OMG's CORBA standard. As a result, the device will be accessible from anywhere in the network regardless of the physical network or software installed. Also included in the system will be a set of objects – predefined interfaces – that specify standard services, such as creating summary reports. These objects then can be modified easily to check for apparent price anomalies so that corrections can be entered on the spot.

In a parallel process, someone from IT will browse through several visual, online libraries to find the mobile, competitive-pricing framework that most closely resembles their planned implementation. Another team member might search for a customer-preference tracking framework. Both frameworks will be customized with self-contained extensions to make them accurately reflect this particular grocery store, the data being gathered and the store's customer profile. This modification will be done simply and quickly using a new class of visual development tools.

The speed advantages of developing applications with OT are important – but far more important will be the synergistic effect of a development environment that builds on the skills and knowledge of both business and technical people.

The speed advantages of developing applications with OT are important – but far more important will be the synergistic effect of a development environment that builds on the skills and knowledge of both business and technical people. Business analysts, line managers and developers will use common tools and languages to analyze and to model the business, how IT is developed and managed, as well as how new applications can be used to make a business more competitive. Evaluating price changes, the financial projections for a new store, the effects of a merger or deployment of a new technology – all the modeling and design can be done using visual tools that group information in a way that emulates the business itself.

5: PSP's Strategy for Object Technology Adoption

5: Summary

- PSP's strategy is to play a leading role in the development of an OT environment that will change the economics and simplify the process of software development for our customers.
- IDC defines 6 major classifications of OT products and services.
- IBM is addressing each of these areas.
- PSP is concentrating on development and delivery of the enabling technology that is fundamental to building the OT industry infrastructure.
- IBM's System Object Model is the cornerstone of IBM's offerings for the emerging multiplatform, distributed object computing environment, and it is available today.
- IBM will deliver an increasingly rich set of application frameworks across the major operating systems in four steps.
- IBM will deliver an efficient OT programming model that enables software developers to produce more customized solutions faster, more economically and with higher quality.

While many of the potential inhibitors of OT have been addressed, a lot of infrastructure still must be built in order to make the promise of this grocery system a reality. ***(See Facing Page)*** IDC defines the following classifications of OT products and services:

- *OO Analysis/ Design Methods and CASE Tools* refer to products that support one or more methods of OO analysis and design.
- *Environments for Development of OO Applications* refer to application development environments that reduce coding. Typically, these are integrated development environments that target client/server applications and use visual programming techniques.
- *Component Software Class Libraries* refer to libraries of reusable objects and frameworks.
- *OO Programming Environments* refer to compilers, programming tools and complete development environments for programmers writing in OO languages like C++ and Smalltalk.
- *Object Database Management Systems* refer to database methods that support object storage and retrieval.
- *Distributed Object Management* refer to the basic system and network services that allow objects to interoperate across a distributed environment.

IBM is addressing each of the elements defined by IDC. In some areas, we have active research and development under way, resulting in specific IBM product offerings. In other areas, we are partnering with other OT leaders to help create a robust set of offerings that support industry standards. This approach is very much in keeping with IBM's OT strategy.

PSP's strategy for Object Technology

PSP's strategy for OT is straightforward: to play a leading role in the development of an OT environment that will change the economics and simplify the process of software development for our customers. ***(See Figure 4)***

Our efforts are guided by four underlying principles:

- We will take a leadership role in research and development.
- We will actively participate in the development and adoption of industry standards.
- We will complement the work of other leading OT companies through key alliances and partnerships.

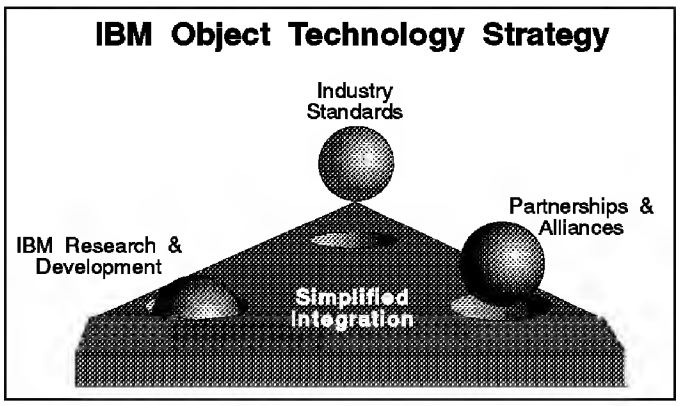


Figure 4 – *These four underlying principles comprise IBM's OT strategy.*

- We will simplify the integration of OT into our customers' operations.

Shaping an Object Technology Infrastructure

When you look around the industry, the collective expertise quickly becomes apparent. A vibrant collection of innovative companies provides languages, applications and tools. Equally apparent, however, is the need for the systems structure and standards that will unify these technologies. This infrastructure must provide an environment for object-based, distributed client/server implementations – implementations that are compatible regardless of the language, platform or network used. PSP is actively developing and delivering the enabling technology fundamental to building that industry infrastructure.

PSP's Object Enabling Strategy

The System Object Model (SOM), the cornerstone for IBM's offerings for the emerging multiplatform, distributed object computing environment, is available today in IBM's SOMobjects™ Toolkit. SOM defines an infrastructure for sharing objects. It allows objects to be packaged in a way that exposes only their interface. As a result, an object can be written in one language and used or refined by another, so, for example, a component written in IBM's C++ is usable and easily refined by Digitalk's Smalltalk. SOM addresses the pervasive need for cross-language support in object development.

In addition, SOM scales up gracefully to support objects that are distributed across a network. The interface for a distributed object is identical to a local SOM object. The SOMobjects Toolkit contains Distributed SOM (DSOM), an ORB that complies with the CORBA standard set by the OMG. DSOM services locate the remote object and route requests and responses, masking this complexity from the developer and end user. Performance is optimized as well. DSOM supports all of the important industry network transports: TCP/IP, Novell's IPX and NetBios.

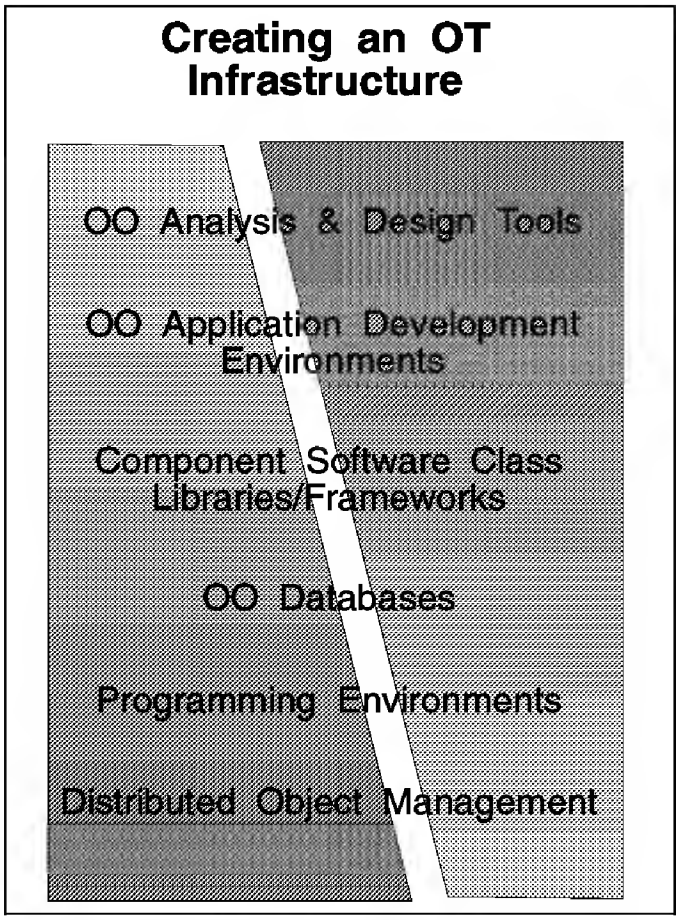


Figure 5 – IBM, through its own alliances and partnerships, as well as other companies, will be delivering products in these categories.

It also will run on the Open Software Foundation's (OSF's) DCE network services when these services become available.

IBM plans to make SOM and its distributed object extensions available on all major platforms. SOM and DSOM are packaged together in a SOMobjects Toolkit which is available today for OS/2® and AIX®. Future availability will include Windows, MVS™, OS/400® and other key industry platforms. The SOMobjects Toolkit will evolve to include new features including evolving CORBA specifications.

SOMobjects is driving some very innovative work that will create a far more powerful and responsive application environment for customers. One example is Component Integration Laboratories (CIL), an industry initiative that has brought

together Apple, Borland, Novell, WordPerfect, IBM and Taligent. Together, these companies are implementing a cross-platform document architecture called OpenDoc. This architecture will simplify creation of compound documents – documents that include text, graphics, images, video and sound, for example. Through OpenDoc, the elements of a compound document can come from a variety of sources, allowing a user to build a powerful multimedia document without having to create and to import the various components, as is the case today. The same technology provides a standards-based way of accessing document components between languages and across networks.

Although OpenDoc may appear similar to Microsoft's Object Linking and Embedding (OLE), industry-standard SOMobjects interfaces ensure that customers will be able to use their favorite application software on their favorite platform. OLE provides this capability only for the Microsoft OLE-enabled applications. OpenDoc will be available during 1994 on multiple platforms: initially Operating System/2 (OS/2), Macintosh System 7, NetWare and DOS/Windows. OpenDoc will interoperate with Microsoft's OLE and the next-generation, document-centered programming model from Taligent.

Distributed Computing Environment (DCE) – As IBM's DSOM and the CORBA specifications evolve, they will address the major issues of networkwide security and directory systems for objects. Similar services are provided today by DCE from the OSF. IBM and Hewlett-Packard are sharing technology so that their ORB will exploit DCE.

Common Operating System Environment (COSE) – IBM is working with SunSoft and Hewlett-Packard to deliver a compatible ORB to the participants in the COSE definition process. Vendors of UNIX-based operating systems, including IBM, Sun Microsystems, Hewlett-Packard, Univel, DEC and SCO, are using the COSE process to define consistent programming interfaces to UNIX-based platforms. The ORB will enable consistent implementation of OT across UNIX-based system environments.

Programming Environment

With the industry's acceptance of SOM, it becomes critical that SOM be incorporated in a new generation of programming languages, compilers and programming tools. A critical mass of OO vendors has embraced SOM and is working to shape a more powerful and flexible OO programming environment:

- IBM and MetaWare will deliver Direct To SOM (DTSOM) C++ compilers. This means these compilers will generate SOM objects automatically.
- Digitalk has announced SOM support in its Smalltalk/V development environment.
- ParcPlace has stated its intention to support SOM in its Smalltalk development environment.
- Watcom has stated its intention to support SOM in its C++ development environment.
- Borland has announced plans to build SOM support into its C++ products.
- MicroFocus has stated its intention to support SOM in its OO COBOL product.

Discussions with other vendors are under way as well.

Object-Oriented Database Management Systems

OO databases are critical to the success of advanced distributed computing environments. A number of companies have focused on this growing segment and have committed to SOM support, including Object Design and Ontos. IBM's strategy for object-oriented database management systems (OODBMS) is twofold:

- In partnership with Object Design, we will provide:
  1. OODBMS services for IBM tools, and
  2. a general OODBMS.
- We also will provide general purpose enabling services for other OODBMS.

Component Software Class Libraries/Frameworks

IBM's SOMobjects Toolkit provides an infrastructure for distributed software components or objects. Frameworks represent groups of compatible software components. In other words, frameworks capture the collected experience of a design team. They address a particular problem and are common in most advanced OT products. Their value is that they automate most of the development effort in a particular area, such as designing and implementing a GUI. IBM will deliver an increasingly rich set of frameworks across key industry operating systems including OS/2, AIX, Windows, the COSE environment and Workplace OS™ in five stages:

1. Frameworks that automate some of the tasks associated with distributed applications.
2. Frameworks that mask the differences between operating systems, allowing applications to be ported easily among them.
3. Taligent desktop frameworks that will radically alter the economics of building advanced graphical applications.
4. Taligent and IBM network frameworks that deal with complex transactions and communications.
5. Taligent and IBM application and system frameworks as Workplace OS – Taligent personality.

These frameworks will simplify deployment of distributed, component-based, collaborative applications. ***(See Figure 6)***

The IBM/Taligent frameworks will provide advanced object application programming interfaces (APIs) for both new and evolving applications. These frameworks will provide the same APIs and advanced functions across all platforms. Developers will be able to adopt the new framework-based services at their own pace. For example, new applications may be written entirely to the object services, while existing applications can take advantage of the new services as they evolve to meet new requirements.

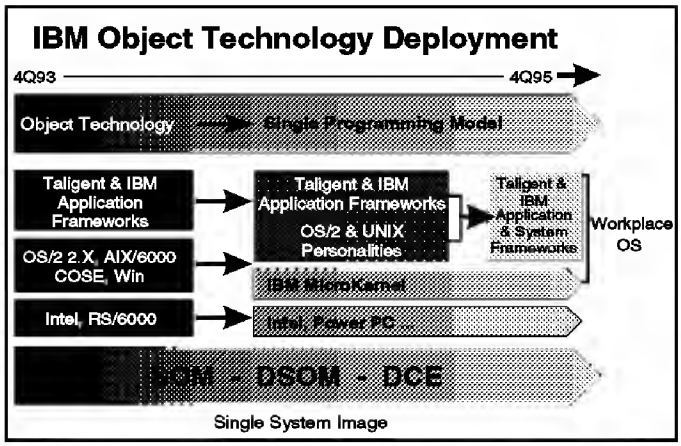


Figure 6 – *IBM will deploy OO technology over existing and evolving industry-leading hardware platforms.*

IBM PSP shipped the SOMobjects Toolkit with several frameworks that help developers build distributed applications. Next, PSP will provide a set of portability frameworks and classes that can substantially isolate a developer from platform differences, everything from memory management to advanced graphics. This is the first set of frameworks that really begins to address the problems of multiplatform object development.

The result of 1994's product deliveries will be a set of complementary frameworks that provide a new programming model for software, an IBM Taligent-based portable application environment. Rather than creating large applications, developers will be able to make small extensions to the frameworks. The environment ensures that these extensions automatically and seamlessly integrate with the framework and each other to exchange data and use each other's services. This portable application environment is an OO development layer that will be made available across all key industry OS platforms including OS/2, Workplace OS, AIX, Windows and others.

Key frameworks in this category will be the document, document user interface and desktop frameworks, as well as a set of generic tools for test, graphics, image editing and database access.

These broad categories of frameworks will allow developers to incrementally evolve their teams and software to take advantage of OT while pre-

When the IBM Taligent-based portable application environment programming model is used, a new style of collaborative, distributed, independently extensible software can be created with a fraction of the programming.

serving their investment in existing procedural application functions and skills.

When the IBM Taligent-based portable application environment programming model is used, a new style of collaborative, distributed, independently extensible software can be created with a fraction of the programming requirement today. This is also the application programming model of the future Taligent OO operating system which extends the use of frameworks onto operating system services. IBM will ship this as the Taligent personality for Workplace OS. All the interfaces defined as part of the portable application environment will be supported in the OO operating system frameworks. Therefore, applications based on the portable application environment can be recompiled to run on any vendor's implementation of the Taligent operating system or other operating system supporting the portable application environment, again preserving the investment in application function. IBM intends to begin rollout of these Taligent frameworks this year.

Object-Oriented Application Development Environments

These environments represent a significant growth area for OO deployment. In this area of OO development environments, IBM is actively working with key independent software vendors (ISV) partners to gain their support for IBM's distributed object computing environment (SOM/DSOM). These partners include vendors such as Digitalk (PARTS), Easel Corporation and Inference Corporation. In addition, IBM has announced VisualAGE, a visual programming environment based on the Smalltalk programming language. This powerful tool, geared primarily for the professional programmer, simplifies the development of client/server applications.

Object-Oriented Analysis and Design Tools

OO analysis and design represents a new and emerging discipline for OO development. It is based on the required transitional shift from existing, proprietary CASE implementations to those based on the object paradigm. IBM's strategy is to continue enabling CASE tools vendors to ensure their tools are able to interact with existing IBM relational database management systems and, over time, to support object-oriented database management systems.

6: Getting Started with Object Technology

6: Summary

- OT gives a powerful new vision of programming. Begin exploiting OT at your own pace.
- The choice of tools varies. There are two distinct tool categories as defined by IDC: OO application development environments and integrated language environments.
- IBM's SOMobjects Toolkit offers advanced development function to users of OO application development environments.
- To prepare to adopt OT, begin with OT education.

Object technology provides a powerful new vision of programming, and it is not too early to begin exploiting it. Using OT does require some training and the adoption of new tools, but implementation can be gradual – starting with a small team of individuals and a simple project. The choice of tools will vary depending on the applications being developed. There are two distinct categories identified by IDC: OO application development environments and integrated language environments. The former is more suitable to groups developing business applications with skills in high-level languages like COBOL. The latter will be more appropriate for groups undertaking system development with lower-level languages like C or C++.

OO application development environments, like but not limited to, IBM's VisualAGE, Digitalk's Smalltalk and PARTS products, Intelligent Environments' Application Manager, ParcPlace and Systems VisualWorks, provide sophisticated tools for building client/server applications that span networks and systems. They focus on powerful tools for developing user interfaces and business logic on PCs and workstations. These tools also provide functions to link to existing transactional and database systems like IBM's Customer Information Control System™ (CICS™) and DB2®. All of these products, and many others, offer several distinct advantages to companies planning for OT. They encourage and reward an objects-based approach to design; they offer an integrated, high-level environment that can speed training; and they work well with existing applications and systems.

Developers using lower-level languages or developing advanced subsystems likely will be better served by the integrated C++ development environments. C++ offers an OO extension to the C language, and both IBM and MetaWare offer versions that provide native support of IBM's SOM. These products combine the development functions for compiling, program editing and debugging in one toolset, but they rely on the developer to provide higher-level constructs or access to other systems and databases.

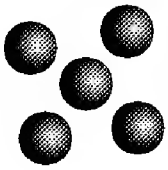
The Smalltalk products from ParcPlace, Digitalk and Easel bridge these two separate worlds and may be the right choice for programmers requiring flexibility – but are unwilling to take on the challenges of lower-level languages like C++. While these environments may not offer the productivity of the application environments, they are more flexible and offer more control over application size and performance.

IBM's SOMobjects Toolkit offers some advanced development functions to users of these environments. The lower-level tools will support SOMobjects and offer programmers access to the detailed function it provides. The application environments generally hide the details of SOMobjects simply surfacing its value by allowing objects to be accessed across a network in a visual development tool.

As the object frameworks are delivered, they will be exploited in this fashion. The underlying details in the language environments are hidden but exploited in the higher-level application tools.

A business's chances of the successful deployment of OT can be increased with OT education. Taligent's advice to developers interested in OT adoption:

- *Learn OO design* – developers who only use C++ as a better C, without fundamentally changing their design approach to use encapsulation, polymorphism and inheritance (all functions of OT), will not realize the substantial benefits of OT.



Vendors Who have Announced Plans to Support SOM

- *American Management Systems*
- *Borland International, Inc.*
- *ChipChat-Cawthon Software*
- *Cirrus Technology, Inc.*
- *Continuum Company, Inc.*
- *Digitalk, Inc.*
- *Easel Corporation*
- *Footprint Software, Inc.*
- *Hewlett-Packard*
- *Inference Corporation*
- *Information Advantage, Inc.*
- *Intermedia Development Company, Inc.*
- *KASEWORKS, Inc.*
- *MetaWare, Inc.*
- *Microformatic*
- *Object Design, Inc.*
- *ParcPlace Systems, Inc.*
- *Raleigh Systems, Inc.*
- *Sundial Corporation*
- *SunSoft, Inc.*
- *Watcom International Corporation*

Figure 7 *– ISVs continue to sign up to support IBM's distributed object computing environment.*

- *Learn an OO language and begin to use it exclusively* – IBM, Intelligent Environments, MetaWare, Digitalk, Easel, ParcPlace, Watcom, Borland and MicroFocus are examples of vendors who offer OO languages and environments and who support or plan to support SOM/DSOM in the near future.

- *Learn to design and work with frameworks* – begin with class libraries and understand both their power and their limitations. Then begin to think in terms of frameworks, becoming familiar with commercially available frameworks from vendors such as Footprint or ChipChat.

Finally, while strong management commitment to OT and careful tools selection are important, consultants and education specialists can be the difference between success and failure. While many specialize in OT, a complete list of educators is well beyond the scope of this paper. However, many OT providers also offer consulting and education, including IBM Consulting Practices, Digitalk, ParcPlace, Knowledge Systems, Raleigh Systems and Andersen Consulting. In addition, IBM Skill Dynamics offers courses in OO concepts, programming, design, analysis, user interface, database and management.

Object Technology Curriculum

Skill Dynamics, an IBM company, offers a comprehensive selection of OT coursework. ***(See Figure 7)*** The OT curriculum and the courses that address each area are outlined on the following page.

If there are OT subjects which you would like to see covered, or if you need a detailed list of course descriptions, pricing and scheduling information, please contact your IBM representative or Skill Dynamics:

Object Oriented Curriculum Manager
Skill Dynamics, An IBM Company
6 East 55th Street
New York, NY 10022
(212) 230-5056

To be added to the OT mailing list, please call (212) 230-5440.

To enroll, call 1-800-IBM-TEACH.

Why wait? Begin to realize the benefits of OT now.

The six categories of OO training with their associated courses are as follows:

Basic Concepts

Basic Concepts, which provides a basic understanding of OO concepts:

Courses:
N1498 – Object-Oriented Software Engineering
Q1022 – Object-Oriented Principles and Their Applications

Prerequisite: Some programming experience.

Note: Object-Oriented Software Engineering includes an introduction to Smalltalk programming.

Management

Management, which provides an understanding of OO management issues:

Courses:
N1605 – Object-Oriented Technology for Managers
N1606 – Object-Oriented Workshop for Managers

Prerequisite: Experience as a manager or team leader for software development.

Analysis Design

Analysis Design, which provides analysis and design skills for OO software development.

Courses:
N1499 – Object-Oriented Design and Use Cases
N1603 – Object-Oriented Analysis and Design
N1604 – Object-Oriented Analysis

Prerequisite: A basic understanding of OO concepts is required; previous analysis and design experience and/or some OO programming experience is recommended.

Note: Some analysis and/or design is included in the courses.
K3612 – Object-Oriented Programming and Design with Smalltalk
N1496 – Object-Oriented Topics with SmallTalk
N1498 – Object-Oriented Software Engineering
N1761 – Introduction to VisualAGE
N1762 – Building Applications with VisualAGE
N1601 – Topics in C++ Programming and Design
N1602 – SOMobjects Developer Toolkit Workshop
Q1073 – Object-Oriented Programming and Design with C++

Languages

Languages, which provides details on OO technical issues as well as use of OT to address important software structuring issues.

Courses:
K3612 – Object-Oriented Programming and Design with Smalltalk
N1496 – Object-Oriented Topics with Smalltalk
N1601 – Topics in C++ Programming and Design
Q1073 – Object-Oriented Programming and Design with C++

Prerequisite: Programming experience. Several months of OO development experience and fluency in Smalltalk or C++.

Database

Database, which provides an understanding of OO DBMS technical issues.

Courses:
N1729 – Object-Oriented Database Management Systems

Prerequisite: Knowledge of database management concepts.

User Interface

User Interface, which provides an understanding of issues, methods and skill requirements for OO user interface design.

Courses:
N1600 – Design of Object-Oriented User Interfaces

Prerequisite: Experience in user interface design and a basic understanding of OO concepts.

Products of other vendors listed in this brochure bear trademarks of their respective vendors.

® AIX, IBM, OS/2, OS/400 and DB2 are registered trademarks of International Business Machines Corporation.

™ SOMobjects, MVS, Workplace OS, CICS and VisualAGE are trademarks of International Business Machines Corporation.

IBM

IBM Corporation
11400 Burnett Road
Internal Zip 2995
Austin, Texas 78758

Produced in the United States of America
11-93

References in this publication to IBM products or services do not imply that IBM intends to make them available outside the United States.